Your search matched **18** of **903905** documents.

A maximum of **18** results are displayed, **25** to a page, sorted by **Relevance** in **descending** order.
You may refine your search by editing the current search expression or entering a new one the text box.
Then click **Search Again**.

brush <and> stroke

Search Again

Results:
Journal or Magazine = **JNL** Conference = **CNF** Standard = **STD**

---

1 **Calligraphic character synthesis using a brush model**
Ip, H.H.S.; Wong, H.T.F.;
Computer Graphics International, 1997. Proceedings , 23-27 Jun 1997
Page(s): 13 -21

[Abstract] [PDF Full-Text (1184 KB)] **IEEE CNF**

---

2 **Digital restoration of painting cracks**
Giakoumis, I.; Pitas, I.;
Circuits and Systems, 1998. ISCAS '98. Proceedings of the 1998 IEEE Internatior Symposium on , Volume: 4 , 31 May-3 Jun 1998
Page(s): 269 -272 vol.4

[Abstract] [PDF Full-Text (580 KB)] **IEEE CNF**

---

3 **Hierarchical classification of paintings using face- and brush stroke mo**
Sablatnig, R.; Kammerer, P.; Zolda, E.;
Pattern Recognition, 1998. Proceedings. Fourteenth International Conference on , Volume: 1 , 16-20 Aug 1998
Page(s): 172 -174 vol.1

[Abstract] [PDF Full-Text (436 KB)] **IEEE CNF**

---

4 **Stroke detection of brush strokes in portrait miniatures using a semi-parametric and a model based approach**
Melzer, T.; Kammerer, P.; Zolda, E.;
Pattern Recognition, 1998. Proceedings. Fourteenth International Conference on ,

Volume: 1 , 16-20 Aug 1998
Page(s): 474 -476 vol.1

[Abstract] [PDF Full-Text (36 KB)] **IEEE CNF**

---

5 **Random paintbrush transformation**
Sziranyi, T.; Toth, Z.;
Pattern Recognition, 2000. Proceedings. 15th International Conference on , Volun
, 2000
Page(s): 151 -154 vol.3

[Abstract] [PDF Full-Text (888 KB)] **IEEE CNF**

---

6 **ArtNova: touch-enabled 3D model design**
Foskey, M.; Otaduy, M.A.; Lin, M.C.;
Virtual Reality, 2002. Proceedings. IEEE , 2002
Page(s): 119 -126

[Abstract] [PDF Full-Text (777 KB)] **IEEE CNF**

---

7 **Visualizing diffusion tensor images of the mouse spinal cord**
Laidlaw, D.H.; Ahrens, E.T.; Kremers, D.; Avalos, M.J.; Jacobs, R.E.; Readhead, (
Visualization '98. Proceedings , 18-23 Oct 1998
Page(s): 127 -134, 527

[Abstract] [PDF Full-Text (3548 KB)] **IEEE CNF**

---

8 **Simulating oriental black-ink painting**
Lee, J.;
Computer Graphics and Applications, IEEE , Volume: 19 Issue: 3 , May/Jun 1999
Page(s): 74 -81

[Abstract] [PDF Full-Text (1224 KB)] **IEEE JRN**

---

9 **A resolution independent nonrealistic imaging system for artistic use**
Kasao, A.; Nakajima, M.;
Multimedia Computing and Systems, 1998. Proceedings. IEEE International Conference on , 28 Jun-1 Jul 1998
Page(s): 358 -367

[Abstract] [PDF Full-Text (1244 KB)] **IEEE CNF**

---

10 **A data structure suitable for representing the calligraphic rules for Chi**

character evaluation
Jianchao Zeng; Inoue, T.; Sanada, H.; Tezuka, Y.;
Pattern Recognition, 1988., 9th International Conference on , 14-17 Nov 1988
Page(s): 181 -183 vol.1

[Abstract] [PDF Full-Text (232 KB)] **IEEE CNF**

---

11 **Development of the Insulator Washer robot**
Hirose, S.; Aoki, S.;
Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conferenc
Volume: 2 , 21-27 May 1995
Page(s): 1783 -1789 vol.2

[Abstract] [PDF Full-Text (1444 KB)] **IEEE CNF**

---

12 **Computer calligraphy-brush written Kanji formation based on the brush-touch movement**
Yamasaki, T.; Hattori, T.;
Systems, Man, and Cybernetics, 1996., IEEE International Conference on , Volum
14-17 Oct 1996
Page(s): 1736 -1741 vol.3

[Abstract] [PDF Full-Text (420 KB)] **IEEE CNF**

---

13 **Forming square-styled brush-written Kanji through calligraphic skill knowledge**
Yamasaki, T.; Hattori, T.;
Multimedia Computing and Systems, 1996., Proceedings of the Third IEEE International Conference on , 17-23 Jun 1996
Page(s): 501 -504

[Abstract] [PDF Full-Text (248 KB)] **IEEE CNF**

---

14 **Using concepts from painting**
Kirby, R.M.; Marmanis, H.; Laidlaw, D.H.;
Visualization '99. Proceedings , 1999
Page(s): 333 -540

[Abstract] [PDF Full-Text (1772 KB)] **IEEE CNF**

---

15 **Adaptive resolution image acquisition using image mosaicing techniqu from video sequence**
Takeuchi, S.; Shibuichi, D.; Terashima, N.; Tominaga, H.;
Image Processing, 2000. Proceedings. 2000 International Conference on , Volume

2000
Page(s): 220 -223 vol.1

[Abstract] [PDF Full-Text (340 KB)] **IEEE CNF**

---

16 **Generating realistic Kanji character images from on-line patterns**
Velek, O.; Cheng-Lin Liu; Nakagawa, M.;
Document Analysis and Recognition, 2001. Proceedings. Sixth International Conference on , 2001
Page(s): 556 -560

[Abstract] [PDF Full-Text (344 KB)] **IEEE CNF**

---

17 **Thinning Arabic characters for feature extraction**
Cowell, J.; Hussain, F.;
Information Visualisation, 2001. Proceedings. Fifth International Conference on ,
Page(s): 181 -185

[Abstract] [PDF Full-Text (404 KB)] **IEEE CNF**

---

18 **Development of a flux compression power unit for millisecond ETC pul power applications**
Goldman, E.B.; Davies, F.; Bickford, K.J.; Smith, E.; Williams, P.; Leone, M.;
Magnetics, IEEE Transactions on , Volume: 35 Issue: 1 , Jan 1999
Page(s): 340 -345

[Abstract] [PDF Full-Text (672 KB)] **IEEE JRN**

A Resolution Independent Nonrealistic Imaging System for Artistic Use

Atsushi Kasao

Graduate School of Information Science and Engineering, Tokyo Institute of Technology, 2-12-1 Ookayama Meguro-ku, Tokyo, 152 Japan

Masayuki Nakajima

Graduate School of Information Science and Engineering, Tokyo Institute of Technology, 2-12-1 Ookayama Meguro-ku, Tokyo, 152 Japan

Abstract

In multimedia environments, digital artwork is usually exhibited on many types of media with different sizes and resolutions. For this reason, it is desirable that artists create works of art in a resolution independent way. In this paper, we present an algorithm for resolution independent nonrealistic imaging. To realize resolution independent imaging, characteristics of all brush strokes are coded and saved in a characteristic table. From the characteristic table, images of any sized and resolution can be created. This characteristic table gives our algorithm the ability to create many types of artistic expression. Three different expressions are exhibited here. In particular the expression in which contours are drawn by pen and ink then colored by brush is , we believe, a new type of expression in computer artwork. Another feature of our algorithm is that the shape of a brush stroke is not set, but is adaptively changed. A less mechanical pattern which seems more natural to humans. As for esthetic evaluation, we do not have a proper evaluation method, but if an image created using this algorithm won an award for digital creation, that would be an indication of the algorithm's success.

1. Introduction

Recently, the chance to see electronic art work through multimedia systems has been increasing. Because each media usually has own resolution, it is better for artists to produce art work in a resolution independent way. We are studying a way to produce a resolution independent automatic imaging system for use by artists.

Many kinds of nonrealistic imaging techniques have been previously presented. Pen-and-ink illustration [1][2] and painterly rendering [3][4] are two representative algorithms. In regard to pen-and-ink illustration, Salisbury et al. [1] proposed a way to produce a same density illustration for any magnification. The conventional painterly algorithm could also be made available in multimedia environments by changing the brush stroke size and reference photograph size in proportion to magnification. Combining the painterly algorithm and multiresolution images [5] is another possibility. However, neither of these methods would work proper at high magnification because a magnified image is warped excessively. A better strategy is to change the imaging algorithm itself to be independent of image size.

Other than these problems, quality and variety of expression have to be considered. Some nonrealistic rendering algorithms [3][4] for animation and real-time processing are available, but we are aiming to produce an automatic nonrealistic imaging algorithm for artistic use.

In the history of art, many new techniques were invented and combined in one image. For example, Albrecht Durer drew "The citadel of Arco in the South Tyrol" [6] by water color and pen. Later Jean Honore Fragonard drew "Scene in a park" by pen-and-brush and brown ink, and watercolor wash. At the very least, imaging algorithm for artists must have the ability to produce many type of expressions.

In the past, we interviewed various artists about what characteristics of a model they pay attention to when they sketch [7]. From these interviews, it was clear that artists grasp different characteristics according to their sketching styles. In an imaging algorithm having a variety of expressions, extracting many characteristics before creating an image is important.

We propose a novel automatic nonrealistic imaging algorithm satisfying the above two characteristics (i.e. independence of image size and extraction of many characteristics before image creation). This algorithm consists of four parts. The first part is segmenting the image, extracting characteristics (average color and shape) from each segment and storing these data into a characteristic table. The second part is arranging the characteristic table. The third part is creating brush strokes from the characteristic table only, and the final part is coloring the brush strokes. In this paper, we show that this algorithm can produce any size image and also can create a special look image painted by brush with contours drawn by pen and ink.

The structure of this paper is as follows: section 2 gives an outline of our clustering algorithm used to produce the characteristic table; section 3 illustrates the basic algorithm for creating a comparatively realistic image; section 4 shows that this basic algorithm can produce any resolution images; section 5 illustrates and discusses how to produce a variety of expressions; and section 6 concludes with final remarks and a discussion of future work.

2. Image segmentation

In general speaking, a nonrealistic rendering image is generated directly from a reference photograph. To produce a

nonrealistic image only from a characteristic table instead of a reference photograph, the segmentation method which segments the photograph must have the ability to create such fine segments that one segment is considered one brush stroke. In addition, extracted characteristics from each segment must be appropriate enough to produce many kinds of artistic images. Using the characteristic table, our first target is to create a realistic rendered image similar to the reference photograph. A realistic rendered image is defined as an image which consists of brush strokes but keeps important impressions of the reference photograph. As pixels in a brush stroke have the same color, the total amount of data is much less than the reference photograph. On the other hand, because this algorithm is not just a lossy compression algorithm, the ability to create a variety of expressions is more important than compression ratio. The more effective the extracted characteristics, the smaller the number of segments required to produce a properly rendered image. The characteristics extraction process proceeds as follows:

1. An reference photograph is segmented into small segments which have neighboring similar color pixels. This segmentation technique must create brush stroke like segments.
2. The characteristics of each segment are entered into a characteristic table.

In the past, we observed how painters sketch a model [7]. The results of these observations gave much useful information for creating our segmentation algorithm.

The observations showed the following expression features: (a) Strokes on contours are very thin. (b) Strokes on details are fine. On the contrary, strokes on coarse areas are large. (c) In fine texture regions like hair, strokes usually do not express hair one by one, but draw an area of hair. The shape of such a stroke is usually stretched along the direction of a texture. (d) If a non-texture area is surround by textured areas, shapes of strokes in the no-texture area are affected by shapes of strokes in the textured areas. These are not strict rules. To produce brush strokes as mentioned above, a 5-dimensional K-means clustering algorithm is adopted. The five dimensions consist of two position parameters and three color parameters, because a brush stroke must be a collection of pixels being near in position and in color space.

Furthermore, to control brush strokes in accordance with above-mentioned rules, the 5-dimensional K-means clustering algorithm must be revised to include information about direction and strength of a texture. In following section, we describe how to calculate the direction and strength of a texture, and how to use this information.

2.1. Notation

We selected the CIELAB color space, because this color space is fit for human vision and the formula for a color difference calculation is easy to define . Here, we define the symbols used in the following sections:

G : A reference photograph
l : L*
a : a*
b : b*
db_n : Edge strength of the nth pixel
da_n : Edge direction of the nth pixel
$Da_{i,j}$: Texture direction of the jth segment at the ith iteration
$Dd_{i,j}$: Directionality strength of the jth segment at the ith iteration
$Dc_{i,j}$: Texture strength of the jth segment at the ith iteration

First, for pixels with color discontinuity, strength and edge direction of the color discontinuity need to be extracted from the reference photograph. The convolutions of the two masks, M_{3-1} and M_{3-2}, shown in Figure 1, with the l component of the reference photograph G are performed respectively. Then, da_n and db_n are calculated using following equations:

$$da_n = \frac{16}{\pi}(\tan^{-1}(\frac{LL}{CC}) + \frac{\pi}{2}) \tag{1}$$

$$db_n = \log(LL^2 + CC^2 + 1) \tag{2}$$

$$CC = M_{3-1} * G \tag{3}$$

$$LL = M_{3-2} * G \tag{4}$$

where * means convolution, and da_n and db_n are quantized and expressed by integers ranging from 0 to 15. The values of da_n express the directions shown in Figure 2.

2.2. Three texture factors extracted from texture in a segment

The K-means algorithm is a method to iterate calculations of a distance between a pixel and the center of a segment successively until the shapes of segments do not change. Accordingly, to give the segmentation algorithm the ability to utilize texture information, the distance function must be modified.

Generally speaking, to perceive some texture, a single pixel is not enough. A small area consisting of some pixels is required. We will show how to extract texture information from da_n and db_n of a segment using the following procedures.

Let i be the index of iteration. First, $N_{0,i,j}$ is defined as the number of pixels which meet the conditions $da_n = 0$ and $db_n > 1$ in the jth segment at the ith iteration. In the same manner, $N_{k,i,j}$ (k = 1~15) are defined. Then, $N_{16,i,j}$ is defined as the number of pixels having no direction (i.e. the number of pixels meeting the condition $db_n \leq 1$ in the jth segment at the ith

M_{3-1}:

-1	-1	-1
0	0	0
1	1	1

M_{3-2}:

-1	0	1
-1	0	1
-1	0	1

Figure 1: The Masks M_{3-1},M_{3-2}

iteration). We have found that the threshold $db \leq 1$ works well. The maximum value from $N_{0,i,j}$ to $N_{15,i,j}$ is chosen. If this value is larger than $N_{16,i,j}/10$, $Da_{i,j}$ is set as the representative direction of the jth segment. On the other hand, if the maximum number is equal to or smaller than $N_{16,i,j}/10$, the number 16 is assigned to $Da_{i,j}$, which means that the jth segment has no texture.

We introduce the measure $Dd_{i,j}$ which indicates how clearly $Da_{i,j}$ represents the direction of a segment. It is assumed that a human is able to discriminate at most two texture directions at a time and if there is a second direction, it must differ from the first direction by about $\pi/2$. Therefore, $Dd_{i,j}$ is expressed by the following equations:

$$Dd_{i,j} = k_2\left(N_{Da_{i,j}} - N^{\perp}_{Da_{i,j}}\right) / \left(N_{Da_{i,j}} + N^{\perp}_{Da_{i,j}}\right) \tag{5}$$

$$N^{\perp}_{Da_{i,j}} = \left(N_{Da_{i,j}+7} + N_{Da_{i,j}+8} + N_{Da_{i,j}+9}\right)/3 \tag{6}$$

where k_2 is a real constant ranging from 0 to 1.

If the result of the calculation of $Da_{i,j}$ is less than 0, $Da_{i,j}$ must be increased by 16 . On the other hand, if the result of calculation of $Da_{i,j}$ is more than 15, $Da_{i,j}$ must be reduced by 16. The threshold, $N_{16,i,j}/10$, is chosen from previous trial experiments.

These two characteristics of texture, $Da_{i,j}$ and $Dd_{i,j}$, are not enough to enable a segmentation algorithm to use texture. As mentioned before, when painters draw fine textures, they are inclined to neglect fine detail and boldly paint the fine texture with broad strokes. However, the bold strokes are usually painted along the direction of the texture. To produce an algorithm which can simulate the features described above, strength of the texture, $Dc_{i,j}$, is necessary to control the parameters of the segmentation. $Dc_{i,j}$ is calculated by the following equation:

$$Dc_{i,j} = k_1 \sum_{k=0}^{15} N_{k,i,j} / N_{i,j} \tag{7}$$

where k_1 is a positive constant.

These three texture factors ($Da_{i,j}$, $Dd_{i,j}$, $Dc_{i,j}$) are used for controlling the distance function in this algorithm.

2.3. K-Means Algorithm with Three Factors of Texture

As was mentioned in the previous section, this algorithm is one of methods used to iterate the calculation of the distance function until the result converges. This segmentation algorithm is executed according to the following steps:

Step 1. Initial arrangement of segment centers

Let K be the initial number of segments. A total of K rectangles are put side by side on an reference photograph G. These rectangles are regarded as the initial segments of G, and all above mentioned characteristics in segments are calculated. The characteristics of the jth segment at the ith iteration ($C_{i,j}$) consist of the center of gravity ($X_{i,j}$, $Y_{i,j}$), the number of pixels ($N_{i,j}$), the average color ($L_{i,j}$, $A_{i,j}$, $B_{i,j}$), the direction of texture ($Da_{i,j}$), strength of directionality ($Dd_{i,j}$) and strength of texture ($Dc_{i,j}$). In short, $C_{i,j}$ consist of ($X_{i,j}$, $Y_{i,j}$, $N_{i,j}$, $L_{i,j}$, $A_{i,j}$, $B_{i,j}$, $Da_{i,j}$, $Dd_{i,j}$, $Dc_{i,j}$). Initial segment characteristics (i.e. $i = 0$) $C_{0,j}$ are calculated and written to the characteristic table. Similar calculations are repeated at each iterations until a stopping condition is satisfied. Assuming that $C_{i,j}$ are being calculated at the ith iteration, we will illustrate following steps .

Step 2. Judging a segment

Pixels are chosen one by one. Then, the weighted Euclidean distance, $H_{i,j,n}$, between $C_{i,j}$ and the chosen nth pixel is calculated by the Equations (8)~(12). First, the distance between the center of gravity and the pixel position, and the color difference between $C_{i,j}$ and nth pixel are calculated.

$$\Delta x_{i,j,n} = (X_{i,j} - x_n) \tag{8}$$

$$\Delta y_{i,j,n} = (Y_{i,j} - y_n) \tag{9}$$

$$\Delta l_{i,j,n} = (L_{i,j} - l_n) \tag{10}$$

$$\Delta a_{i,j,n} = (A_{i,j} - a_n) \tag{11}$$

$$\Delta b_{i,j,n} = (B_{i,j} - b_n) \tag{12}$$

Second, the weighted distance between the center of gravity and the pixel's position is calculated, taking the texture into account.

$$\Delta x'_{i,j,n} = \mathrm{F}(\Delta x_{i,j,n}, \Delta y_{i,j,n}, Da_{i,j}, Dd_{i,j}, Dc_{i,j}) \Delta x_{i,j,n} \tag{13}$$

$$\Delta y'_{i,j,n} = \mathrm{F}(\Delta x_{i,j,n}, \Delta y_{i,j,n}, Da_{i,j}, Dd_{i,j}, Dc_{i,j}) \Delta y_{i,j,n} \tag{14}$$

where F(•) is the function to control the shape and size of

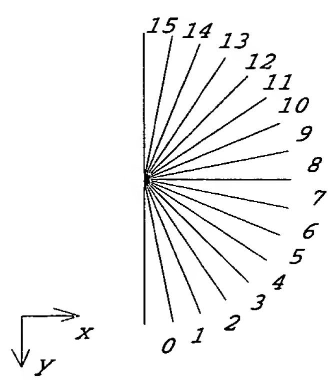


Figure 2: The directions from 0 to 15

segments.

If $\left| Da_{i,j} - (\tan^{-1}(\frac{-\Delta y_{i,j,n}}{\Delta x_{i,j,n}}) + \frac{\pi}{2}) \frac{16}{\pi} \right| < 8$

then

$$F(\Delta x_{i,j,n}, \Delta y_{i,j,n}, Da_{i,j}, Dd_{i,j}, Dc_{i,j}) = 1 + Dc_{i,j}$$

$$\times \{1 + Dd_{i,j} \left| Da_{i,j} - (\tan^{-1}(\frac{-\Delta y_{i,j,n}}{\Delta x_{i,j,n}}) + \frac{\pi}{2}) \frac{16}{\pi} \right| /4 - 1\} \quad (15)$$

If $\left| Da_{i,j} - (\tan^{-1}(\frac{-\Delta y_{i,j,n}}{\Delta x_{i,j,n}}) + \frac{\pi}{2}) \frac{16}{\pi} \right| \geq 8$

then

$$F(\Delta x_{i,j,n}, \Delta y_{i,j,n}, Da_{i,j}, Dd_{i,j}, Dc_{i,j}) = 1 + Dc_{i,j}$$

$$\times \{1 + Dd_{i,j}(16 - \left| Da_{i,j} - (\tan^{-1}(\frac{-\Delta y_{i,j,n}}{\Delta x_{i,j,n}}) + \frac{\pi}{2}) \frac{16}{\pi} \right| /4 - 1)\} \quad (16)$$

These equations are combined, and $H_{i,j,n}$ is defined as

$$H_{i,j,n} = k_x \Delta x'^{2}_{i,j,n} + k_x \Delta y'^{2}_{i,j,n} + k_l \Delta l^{2}_{i,j,n} + k_a \Delta a^{2}_{i,j,n} + k_b \Delta b^{2}_{i,j,n} \quad (17)$$

where k_x, k_y, k_l, k_a, k_b are constants.

The distance $H_{i,j,n}$ between all segments and the nth pixel are calculated and the nth pixel is included in the jth segment with the smallest value of $H_{i,j,n}$. Figure 3 shows the relationship between the vector from the center of gravity $(X_{i,j}, Y_{i,j})$ to the nth pixel and the direction of texture in a segment $Da_{i,j}$. If the direction of this vector comes close to $Da_{i,j}$, the value of $H_{i,j,n}$ is decreased. On the other hand, if the direction differs from $Da_{i,j}$ by $\pi/2$, the value of $H_{i,j,n}$ is increased. As a result, the jth segment includes many pixels in the directions close to $Da_{i,j}$ or $Da_{i,j} + \pi$, but includes few pixels in the other directions.

Furthermore, if almost all pixels in the jth segment have similar values to da_n and satisfy the condition $db_n > 1$, then $Dd_{i,j} \approx k_2$, and the shape of the segment is stretched to be a narrow ellipse. On the contrary, if many pixels in the jth segment satisfy only the condition, $db_n > 1$, regardless of the direction, then $Dc_{i,j} \approx k_1$. Eventually, the size of the jth segment becomes small and the contour of a segment becomes simple. This fact is because of the following reason: if the value of $Dc_{i,j}$ is close to k_1, F(•) becomes large. These calculations are executed successively for every segment.

Step 3. Recording to the characteristic table

$C_{i,j}$ in all newly calculated segments are written over $C_{i-1,j}$ in the characteristic table.

Step 4. Checking the stopping condition

The discrepancy $H'_{i,j}$ between $C_{i,j}$ and $C_{i-1,j}$ is calculated by the following equations:

$$\Delta_{X,i+1,j} = (X_{i+1,j} - X_{i,j}) \quad (18)$$

$$\Delta_{Y,i+1,j} = (Y_{i+1,j} - Y_{i,j}) \quad (19)$$

$$\Delta_{L,i+1,j} = (L_{i+1,j} - L_{i,j}) \quad (20)$$

$$\Delta_{A,i+1,j} = (A_{i+1,j} - A_{i,j}) \quad (21)$$

$$\Delta_{B,i+1,j} = (B_{i+1,j} - B_{i,j}) \quad (22)$$

and

$$H'_{i+1,j} = k'_x (\Delta^2_{X,i+1,j} + \Delta^2_{Y,i+1,j}) + k'_l \Delta^2_{L,i+1,j} + k'_a \Delta^2_{A,i+1,j} + k'_b \Delta^2_{B,i+1,j} \quad (23)$$

The above steps are repeated until $H'_{i,j}$ has a smaller value than a previously chosen value (Cd_1). We have found that a value of $Cd_1 = 1$ works well. When this iterative process stops, the segmentation is completed.

As this new 5-dimensional K-means clustering algorithm is a very time consuming process, we chose to divide the reference photograph into small rectangular regions and apply the algorithm to each region separately. Figure 4 shows the rectangular areas consisting of a core area and overlapping areas [8]. The overlapping areas surround the core area. The new clustering algorithm is applied to each rectangular area. Without overlapping areas, each small rectangular area would be clustered independently. That would cause a discontinuity of a segment on the border. However, if there are overlapping areas, these are clustered several times. This procedure decreases the defects of discontinuity caused by

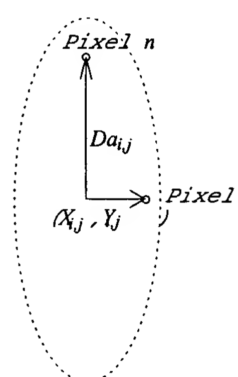


Figure 3: Positional relationship between the pixel n and the center of gravity of the jth segment $(X_{i,j}, Y_{i,j})$

separately applying the K-means clustering algorithm to small areas.

This explanation outlines our new clustering algorithm. Regarding this algorithm, deciding a suitable number of segments at the beginning is difficult. However, this algorithm can segment each small area according to different conditions. The conditions are changed while this algorithm is repeatedly apply to the same small rectangular area. Accordingly, the number of segments can be increased until the maximum color deviation of pixels in a segment is within a certain value (Cd_2). We have found that a value of $Cd_2 = 20$ works well. This segment increasing process efficiently produces enough segments for realistic imaging. Finally, when whole process is finished, the group of characteristics ($C_{i,j}$) is written to the corresponding cell (C_j) in the characteristic table. In the following sections, we will call this algorithm the adaptive K-means method.

3. The basic algorithm for a realistic image

3.1. Additional characteristics

As mentioned in the previous section, C_j consists of (X_j, Y_j, N_j, L_j, A_j, B_j, Da_j, Dd_j, Dc_j). C_j is not enough to create brush strokes for a realistic image, because it does not contain segment shape information. Assuming that each brush strokes is an ellipse, let D_j be the direction of the long axis and let H_j be the oblate rate. These two value are added to C_j. D_j and H_j are calculated from the deviations of pixels in the segment. First, the dispersion in relation to the x-directional deviation from the segment center is defined by dx_j, the y-directional dispersion is defined by dy_j, and the co-dispersion is defined by k_j. Furthermore, let the first and second primal element be L_{1j} and L_{2j} respectively. These two values are calculated from the above values. D_j and H_j are then obtained as followers:

$$D_j=\tan^{-1}\left(\frac{-W_{1j}}{W_{2j}}\right)/\pi+\frac{1}{2} \tag{24}$$

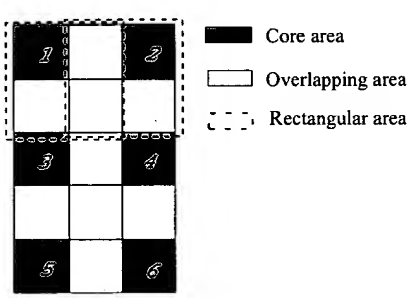


Fig. 4: An example of an image divided into many small areas.

$$H_j=(L_{1j}-L_{2j})/(L_{1j}+L_{2j}) \tag{25}$$

where

$$W_{1j} \quad \frac{k_j}{\sqrt{k_j^2+(L_{1j}-dx_j)^2}} \qquad \text{If } \sqrt{k_j^2+(L_{1j}-dx_j)^2} \quad 0,$$

$$W_{1j} \quad 0 \qquad \text{If } \sqrt{k_j^2+(L_{1j}-dx_j)^2} \quad 0, \tag{26}$$

$$W_{2j} \quad \sqrt{1 \quad |W_{1j}^2} \tag{27}$$

Finally, D_j, H_j are added to C_j in the characteristic table

3.2. The basic imaging algorithm

The reference photograph and the results of segmentation are shown in Figures 5(a) and 5(b) respectively. Figure 5(c) is produced by painting over every segment by the average color of that segment This image resembles the reference photograph and the shapes of almost all segments are ellipses, similar to brush strokes. Therefore, the adaptive K-means method is considered to work well.

In order to reproduce the shapes of segments in Figure 5(c) from only the characteristics of the characteristic table, we need a reproduction process. The following steps show this reproduction process.

[Step 1] Assign the pixel numbers

This method can produce an image of any size. For simplicity, we will make the image the same size as the reference photograph.

In the first row of the image, pixels are assigned serial numbers from the left side to the right side. In the next row, pixels are assigned serial numbers from the right side to the left side. In this manner, the pixels in odd rows of the reference photograph are assigned serial numbers from left to right, and pixels in even rows are assigned serial numbers from right to left. This way of numbering distributes all serial numbers on the image side by side.

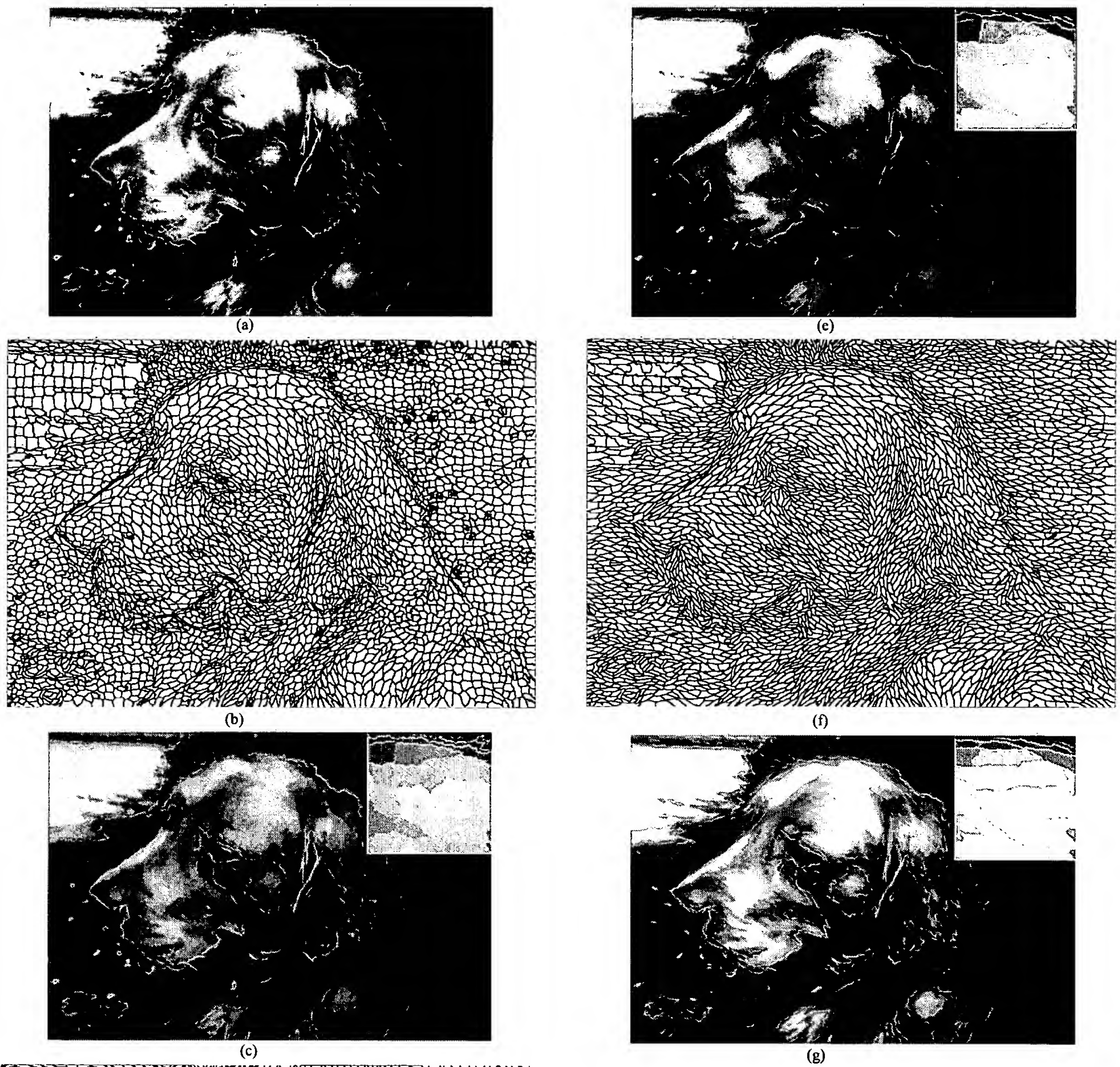
[Step 2] Create the neighboring segments table

For every segment, the nearest 30 segments are chosen and these segments are written to the neighboring segments table shown in Table 1 in order of shortest distance.

[Step 3] Creating the brush strokes

A brush stroke j on a new image is created from a corresponding segment j on a segmented image. In the same way as the previously shown segmentation, the position of a brush stroke is indicated on the new image by a corresponding segment number. Because pixel n is next to pixel n-1, if pixel n-1 is a member of brush stroke j, pixel n must be in one of the nearest 30 segments of segment j.

Accordingly, after calculating the distances between the nearest 30 neighbors and pixel n by equation (28), the pixel n is chosen as a member of the brush stroke which has the smallest M_j (i.e. the shortest distance) .



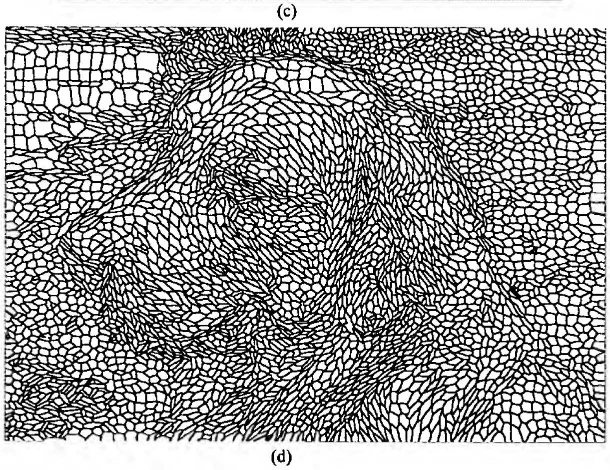


Figure 5: Brush strokes made from an photo image: (a) the original image, (b) the segmented image, (c) an image colored by the average colors of segments in (b), with a magnified insert, (d) brush strokes made from a characteristic table, (e) an image painted by the average colors of brush stokes in (d), with a magnified insert, (f) brush strokes made from a revised characteristic table, and (g) an image painted by average the colors of brush stokes in (f), with a magnified insert.

$$M_j = \frac{\{(x_n - X_j)^2 + (y_n - Y_j)^2\}(1.2 + f'_j H_j)^2}{N_j} \tag{28}$$

$$f'_j = 2 \cdot f_j - 1 \tag{29}$$

$$\text{If} \left| \tan^{-1}\left(\frac{Y_j - y_n}{x_n - X_j}\right)\frac{1}{\pi} + \frac{1}{2} - D_j \right| \quad \frac{1}{2}$$

$$f_j = 2\left| \tan^{-1}\left(\frac{Y_j - y_n}{x_n - X_j}\right)\frac{1}{\pi} + \frac{1}{2} - D_j \right| .$$

$$\text{If} \left| \tan^{-1}\left(\frac{Y_j - y_n}{x_n - X_j}\right)\frac{1}{\pi} + \frac{1}{2} - D_j \right| \quad \frac{1}{2}$$

$$f_j = 2\left\{ 1 - \left| \tan^{-1}\left(\frac{Y_j - y_n}{x_n - X_j}\right)\frac{1}{\pi} + \frac{1}{2} - D_j \right| \right\} \tag{30}$$

As can be seen, in equation (28), M_j is inversely proportional to N_j. For this reason, a pixel is apt to be a member of the segment j with a large N_j. When $H_j \neq 0$, pixels in the direction D_j from the center of the segment j are more likely to be a member of the segment j than pixels in other directions. The above method makes brush strokes similar to the segments on the just segmented image. When this procedure is performed for every pixel, Figure 5(d) is produced.

[Step 4] Coloring process

This step is to paint each brush stroke with its average color. Figure 5(e) is produced by painting over every brush stroke of Figure 5(d) by the average color of that segment.

3.3. The propagation algorithm of characteristics

Figure 5(e) is similar to Figure 5(c) in that both of the figures have the previously mentioned desired expression features (a) ~ (c) (see the introduction of section 2). However, their brush stroke gives a mechanical feeling because brush strokes on the area without texture are simple rectangles. In other word, their brush strokes do not have the expression feature (d). The expression feature (d) is that if the non-texture area is surround by texture areas, the shape of the stroke on the non-texture area is affected by the shape of strokes on the surrounding texture area. We will show how to add this feature to Figure 5(e). From experience, if H_j is larger than 0.7, the shape of the brush stroke resembles a real brush stroke. If H_j is not larger than 0.7, the top 10 largest H_k from the segments in the neighbor segments table are chosen. Then, H_k which meet the condition, $H_k > H_j$, are averaged, and this value is substituted for H_j. In the same way, D_j is replaced for the average. If this process is repeated many times, the influence of a brush stroke with large H_j propagates to other brush strokes far away. From the results of previous experiments, we decide to iterate this process 100 times. Brush strokes created by using this revised characteristic table are shown in Figure 5(f), and Figure 5(g) is the image which is painted using the average color of each brush stroke in Figure 5(f). It is recognized that the brush strokes of Figure 5(f) are much more proper for painterly imaging than Figure 5(d).

Table 1: The Neighbor Segments Table

Segment	the nearest 30 segments
1	1, 8,10, 5, 3, 7, 2, • • • ,20
2	2, 3, 6, 5, 8, 4, 9, • • • ,16
3	3, 2,16, 5, 4, 6, 9, • • • ,18
4	4, 9, 3,12, 8, 1, 2, • • • ,25
5	5, 2, 21,7, 3, 7, 9, • • • ,34
•	• • • • • • • • • • •
•	• • • • • • • • • • •
•	• • • • • • • • • • •

4. Resolution Independent Imaging

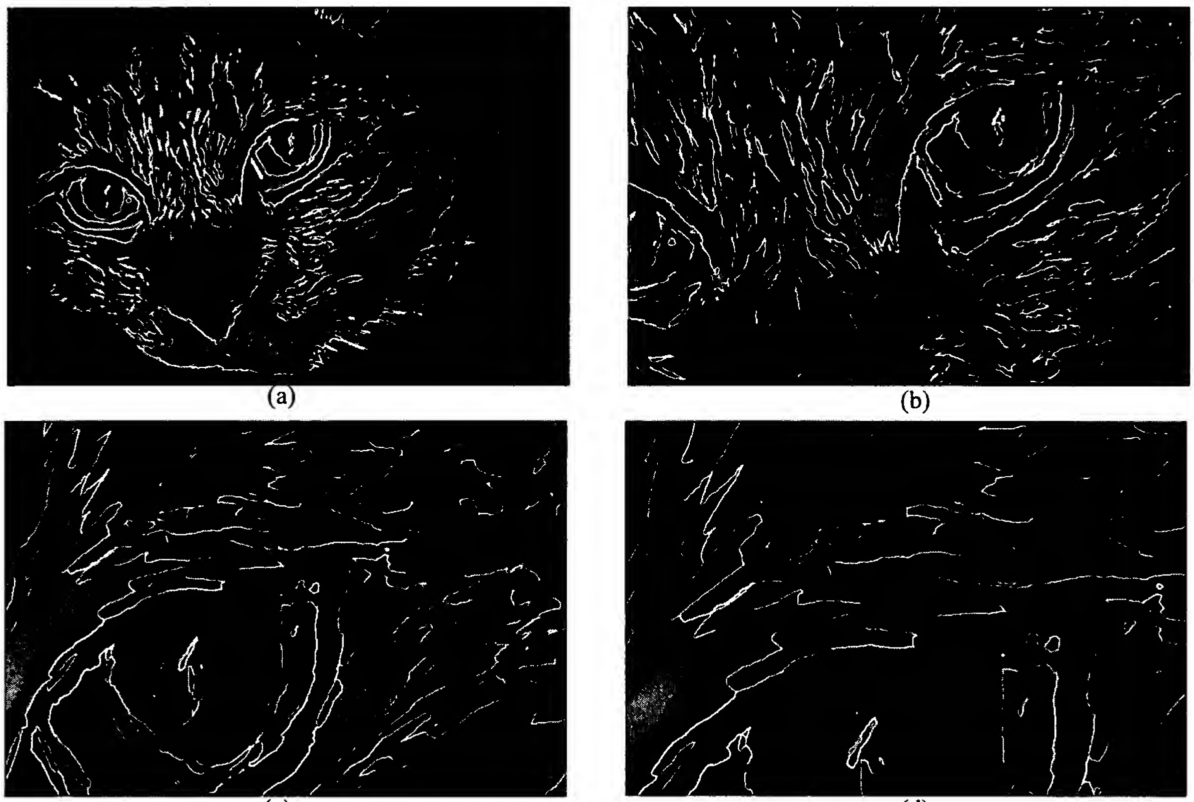
In Figure 6, four different-sized images are shown. All images were produced by our imaging algorithm. Because this algorithm produces the painterly image using only the characteristic table, however, high resolution or any large-sized images can be created by using the same algorithm. In Figure 6, all images consist of 1218×815 pixels. Figure 6(a) is the whole image, Figure 6(b) is part of a 2436×1630 (2×) image, Figure 6(c) is part of a 3654×2445 (3×) image, and figure 6(d) is part of a 4872×3260 (4×) image. None of these image show any jagged edges.

5. Brush expression with contours by pen and ink and a variety of other expressions

In previous sections, It was shown that this algorithm had the ability to produce any size painterly image using only the characteristic table. However, if this algorithm could not create a variety of expressions, it would be useless. This algorithm has mainly three ways to add different expressions to the basic painterly image (i.e. Figure 5 (e)). One way is to modify step 3 of the basic imaging algorithm. Another one is in step 4, and the last place is in relation to the propagation algorithm of the characteristic table. The last method is used for propagating conspicuous characteristics to surrounding segments. In Figure 5(e) shapes of brush strokes were propagated, but color can also be propagated. We will illustrate two new expressions by modifying step 2 and step 3, and the previously mentioned hybrid expression painted by brush with contours drawn by pen and ink.

5.1 Expression with impressionist-like brush strokes

The shape of a brush stroke varies by modifying the equation (28). To make the shape of the brush stroke fine and



(a) (b) (c) (d)

Figure 6: The results of resolution independent imaging

Figure 7: Expression with impressionist-like brush strokes.

Figure 8: Special expression with color gradation within brush strokes.
('97 Toray Creation Awards, Exereses Award)

Figure 9: An impressionist style painting with contours by pen and ink.

long, equations (28) ~ (30) are replaced by equations (31) ~ (33).

$$M_j = \frac{\left\{(x_n - X_j)^2 + (y_n - Y_j)^2\right\}(1.2 + f'_j H'_j)^2}{N_j} \tag{31}$$

$$f'_j = \frac{2}{1+\exp(7(0.3-f_j))} - 1 \tag{32}$$

$$H'_j = \frac{1}{1+\exp(7(0.5-H_j))} \tag{33}$$

The color of brush strokes can be changed by varying L_j, A_j, and B_j of the characteristic table by a random number ranging between ±20. This color modification can add to the image a touch of impressionist painting. The result is shown in Figure 7.

5.2 Special expression with color gradation within brush strokes

This algorithm can create two types of color gradation in brush strokes. The shape of a brush stroke of Figure 8 is the same as that of Figure 5(g). This image was produced with the following conditions: if the value of H_j is under 0.9, the segment j is colored by a star shaped gradation. On the other hand, if the value of H_j is more than 0.9, this brush stroke is colored by gradation along the direction of the normal to D_j.

5.3 Brush expression with contours by pen and ink

As was previously mentioned, besides creating a variety of expressions, expressing different techniques on one artwork at a time is important. Figure 9 seems to have strokes painted by brush with pastel colors and contours by pen and ink. In this image a contour line is a segment which meets the condition ($H_j > 0.85$) being made very slender and colored black. When $H_j \leq 0.85$, equation (28) and (29) are used for producing the usual brush strokes. On the other hand, when $H_j > 0.85$, (28) and (29) are substituted by following equations to turn the segments into contour lines.